On the Improvement of Classifying EEG Recordings Using Neural Networks

Yiran Zhao
University of Illinois at Urbana-Champaign
zhao97@illinois.edu

Shuochao Yao
University of Illinois at Urbana-Champaign
syao9@illinois.edu

Shaohan Hu
IBM Research
shaohan.hu@ibm.com

Shiyu Chang
IBM Research
shiyu.chang@ibm.com

Raghu Ganti
IBM Research
rganti@us.ibm.com

Mudhakar Srivatsa
IBM Research
msrivats@us.ibm.com

Shen Li
IBM Research
shenli@us.ibm.com

Tarek Abdelzaher
University of Illinois at Urbana-Champaign
zaher@illinois.edu

***Abstract*—This paper presents improved results on classifying electroencephalography (EEG) recordings using deep learning. The task is to classify movements that the subject is thinking about (motor imagery), using only the recorded electrical activities on the scalp. The challenges are: poor signal-to-noise ratio; interference from numerous sources such as electrical line noise, muscle activity, and eye movements; considerable variability between individuals and even recording sessions. Traditional signal processing techniques such as frequency band analysis, common spatial pattern (CSP) algorithm or independent component analysis (ICA) fall short due to their limited capacity. Thanks to the rise of big data in healthcare, medical recordings now come in abundance. Therefore deep learning which relies on large amounts of training data is becoming the new cutting edge tool. We present a significant improvement of classification accuracy on the Brain-Computer Interfaces Competition IV dataset (2a), and compare the results of various state of the art neural network structures.**

***Keywords*-EEG; deep learning; BCI Competition IV; motor imagery;**

I. Introduction

EEG based systems have gained popularity in clinical applications due to their advantage of being non-invasive. They enable paralyzed patients to communicate and control robots. However, their effectiveness is limited by many daunting challenges such as a low signal-to-noise ratio and strong interference. Traditional methods such as frequency band analysis [1], ICA [2], and CSP algorithm [3] are susceptible to such interference and therefore difficult to achieve good accuracy.

Recently, with the increasing abundance of data, deep neural network (DNN) fits in as an end-to-end model that does not require any handcrafted feature or feature selection. Convolutional neural networks (CNN) have been successfully applied to image classification tasks [4], achieving better results than human beings. Recurrent neural networks (RNN) such as Long Short Term Memory [5] have the ability to capture long-term dependencies. A combination of CNN and RNN across multiple input modalities with compressing ability has been exploited to improve the results of classification tasks and regression tasks [6], [7]. Therefore, the availability of EEG data has tempted many researchers to use DNN. For the BCI Competition IV dataset 2a, there are 4 classes, 25 channels, 250 samples per second and 3 seconds per trial. Existing work mostly applies CNN directly on the "image" of size 25 by 750. Although using CNN [8] indeed achieves better accuracy (73.7%) than traditional signal processing techniques (68.0%), our improved data processing and DNN structure can yield significantly better classification accuracy (81.1%).

II. Related Work

The BCI Competition IV [9] winner and several top teams all adopt CSP algorithm on bandpass-filtered data to generate features, and then use SVM, LDA or Bayesian methods to train a classifier. Their best accuracy is 67.7%. Ang et al. [10] propose FBCSP method which enhances the performance of the original CSP algorithm by performing autonomous selection of discriminative subject-specific frequency range for band-pass filtering, which yields an accuracy of 68.0%.

More recently, [8] proposes the combination of a shallow 2-layer CNN and a deeper 5-layer CNN model, which achieves an accuracy of 73.7%. While [8] outperforms the FBCSP method, it also shows that the more advanced residual networks [11] do not yield as good accuracy as shallow CNN, possibly due to limited data size. In our work, we add a fully connected (FC) layer before CNN projecting the 25-channel input non-linearly onto a slightly larger dimension, which leads to much better results.

Some papers divide the whole trial into several frames and use RNN to connect the output of each frame, which is similar to DeepSense [6]. We compare with a similar DNN structure called *CNN+RNN* that applies CNN to each frame, and then applies RNN (GRU) to the output of CNN to learn the cross-frame relationship. Another recent RNN model is Clockwork RNN [12], where the hidden recurrent units are divided into different groups, each being updated in a different frequency. The Clockwork RNN has shown to be able to capture both short term and long term temporal relationships. We implement an improved version of Clockwork RNN on top of CNN (called *CNN+CW*) and show that its performance is comparable to traditional methods but not so good as our best CNN model (named *CNN++*).

978-1-5386-2715-0/17/$31.00 ©2017 IEEE

III. Design

A. Dataset overview

The dataset contains 2592 training and 2592 test recordings of 4-class motor imagery (left hand, right hand, feet, tongue) from 9 subjects, each performed 576 trials. The raw recordings are from 22 EEG channels and 3 EOG channels, sampled at 250 Hz. The recordings are bandpass-filtered between 0.5 Hz and 100 Hz, and an additional 50 Hz notch filter is applied to suppress power line noise.

B. Data processing

We first apply band-pass filter with frequency between 1 Hz and 40 Hz. Then we down sample the recordings to reduce sample rate from 250 Hz to 125 Hz, while at the same time double the data size. We do not apply data normalization since we observe that un-normalized recordings yield about 1% better accuracy. We do not apply data augmentation such as adding noise because it did not improve the performance probably due to already low signal-to-noise ratio. After pre-processing, there are about 5184 3-second training trials.

C. Neural network models

We propose our CNN++ model and compare with CNN+RNN and CNN+CW models. We use $conv([N_i, H, W, N_o], [S_h, S_w])$ to denote a convolutional layer with N_i input channels and N_o output channels (filters), with filter size (H, W) and stride (S_h, S_w). And $pool([H, W], [S_h, S_w])$ denotes a max pooling layer with window size (H, W) and stride (S_h, S_w).

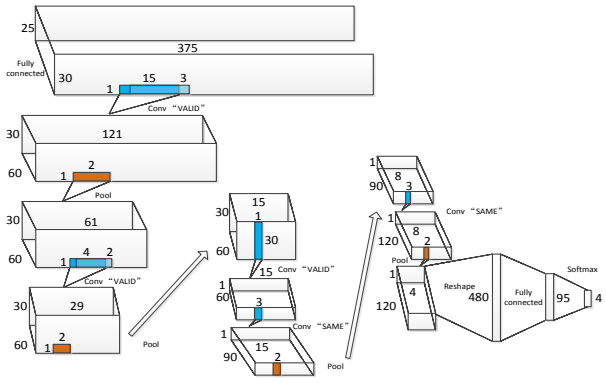

Figure 1. CNN++ structure.

1) CNN++: We call our model CNN++, which consists of 5 CNN and max pooling layers with an input fully connected (FC) layer as shown in Figure 1. Common Spatial Pattern (CSP) algorithm linearly projects the input channels onto fewer channels so that the features (e.g. signal variance) of the projected channels between different classes are the most distinguishable. Inspired by CSP, we find that a simple input FC layer that projects 25 channels into 30 channels works the best. The projected input of each trial has size 30×375, which is then fed to the convolutional layers. The first CNN layer applies $conv([1, 1, 15, 60], [1, 3])$ and $pool([1, 2], [1, 2])$. The second layer applies $conv([60, 1, 4, 60], [1, 2])$ and $pool([1, 2], [1, 2])$. The third layer applies $conv([60, 30, 1, 60], [1, 3])$ without pooling. The fourth layer applies $conv([60, 1, 3, 90], [1, 1])$ and $pool([1, 2], [1, 2])$. The last layer applies $conv([90, 1, 3, 120], [1, 1])$ and $pool([1, 2], [1, 2])$. The output is followed by a FC layer with output dimension 95. Finally a softmax layer is added, yielding the probability of the 4 classes. All convolutional layers have dropout [13] with probability 0.5 and also batch normalization [14].The activation function is Exponential Linear Units (ELU).

2) CNN+RNN: The input trial of 375 time steps is divided into 5 frames, each containing 75 steps. We apply the same input FC layer as our CNN++ model. The convolutional filters are shared among all the frames, where each frame has size 30×75. We found that recurrent layers can increase the likelihood of overfitting, so after tuning we adopt slightly few filters for each layer. The first layer applies $conv([1, 1, 6, 40], [1, 1])$ and $pool([1, 2], [1, 2])$. The second layer applies $conv([40, 1, 4, 40], [1, 1])$ and $pool([1, 2], [1, 2])$. The third layer applies $conv([40, 30, 1, 60], [1, 1])$ without pooling. The fourth layer applies $conv([60, 1, 3, 80], [1, 1])$ and $pool([1, 2], [1, 2])$. The last layer applies $conv([80, 1, 3, 80], [1, 1])$ and $pool([1, 2], [1, 2])$. The output of the 5 frames are fed to a 2-layer GRUs [15] with hidden dimension 80. The average of the hidden states from 5 frames are followed by a softmax layer. Dropout and batch normalization are applied to both CNN and RNN layers.

3) CNN+CW: The input of size 25×375 is projected to size 30×375 by a FC layer. A 4-layer CNN and max pooling are applied. To prevent overfitting, we use fewer filters. The first layer applies $conv([1, 1, 6, 20], [1, 1])$ and $pool([1, 2], [1, 2])$. The second layer applies $conv([20, 1, 3, 20], [1, 1])$ and $pool([1, 2], [1, 2])$. The third layer applies $conv([20, 30, 1, 20], [1, 1])$ without pooling. The fourth layer applies $conv([20, 1, 3, 20], [1, 1])$ with no pooling. The output is a 1×90 "image" with depth 20, and is reshaped to 20×90. Then a Clockwork RNN is applied.

Clockwork RNN [12] is essentially a simple recurrent network but has its hidden cells divided into multiple groups, each being updated with a different period. For example, if there are 200 hidden units and 5 groups, then each group has 40 units. The update period of each group is exponentially increasing, i.e., the first group updates every 1 time step, the second updates every 2 steps, the third every 4 steps, the fourth every 8 steps and the fifth every 16 steps. By updating at different clock rates, it can capture both long term and short term relations. The original work divides the hidden units into equal-size groups, we found that in our case, having more units into faster groups yields better results. So we divide 200 hidden units unequally into 6

groups, with updating period 1, 2, 4, 8, 16, 32. The 6 groups have size 76, 38, 38, 19, 19, 10. The final state of the 200 hidden units are connected to a FC layer of size 100, which is then followed by the softmax layer. Dropout and batch normalization is applied to only the CNN part.

IV. Experiments

We use the same training configuration for all models, where softmax cross entropy is the loss function and L2 regularization is applied. The number of epochs is 100, and batch size is 50. We use RMSProp optimizer with exponentially decaying learning rate from 0.01 to 0.001. The testing results are shown in Table I.

Table I
Classification accuracy on test data.

CNN++	CNN+RNN	CNN+CW	Previous best [8]
81.1%	69.1%	68.0%	73.7%

From Table I we can see that CNN+CW achieves the same accuracy as the baseline FBCSP [10] method, while both CNN+RNN and CNN+CW yield less accuracy than pure CNN models (CNN++ and previously best).

In addition, without input FC layer, all three models show around 3% lower accuracy. It is because the input non-linear projection automatically learns the spatial patterns as CSP algorithm does. The difference is that CSP maps the input channels onto much fewer channels, while the FC layer maps to more channels (30 in our case).

Another observation is that RNN is not necessarily a good practice. In our dataset, the trials are only 3 seconds long, so there is really not much "long term" relations to learn. On the other hand, CNN can learn the local time and frequency features while at the same time learn more abstract features in higher layers which covers more time steps.

V. Conclusion

This paper presents an improved version of CNN model that yields 13.1% higher accuracy than the best traditional signal processing method called FBCSP, and has 7.4% higher accuracy than the existing best deep learning based method. We also implement other two state of the art recurrent network models and compare with their performance. We find that by adding an input fully connected layer and using only CNN lead to the best result.

Acknowledgment

This research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-16-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Army Research Laboratory, the U.S. Government, the U.K. Ministry of Defence or the U.K. Government. The U.S. and U.K. Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] J.-A. Martinez-Leon, J.-M. Cano-Izquierdo, and J. Ibarrola, "Feature selection applying statistical and neurofuzzy methods to eeg-based bci," *Computational intelligence and neuroscience*, vol. 2015, p. 54, 2015.

[2] T.-W. Lee, M. Girolami, and T. J. Sejnowski, "Independent component analysis using an extended infomax algorithm for mixed subgaussian and supergaussian sources," *Neural computation*, vol. 11, no. 2, pp. 417–441, 1999.

[3] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller, "Optimal spatial filtering of single trial eeg during imagined hand movement," *IEEE transactions on rehabilitation engineering*, vol. 8, no. 4, pp. 441–446, 2000.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "Deepsense: A unified deep learning framework for time-series mobile sensing data processing," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017.

[7] S. Yao, Y. Zhao, A. Zhang, L. Su, and T. Abdelzaher, "Deepiot: Compressing deep neural network structures for sensing systems with a compressor-critic framework," in *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*. ACM, 2017.

[8] R. T. Schirrmeister, J. T. Springenberg, L. D. J. Fiederer, M. Glasstetter, K. Eggensperger, M. Tangermann, F. Hutter, W. Burgard, and T. Ball, "Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human eeg," *arXiv preprint arXiv:1703.05051*, 2017.

[9] C. Brunner, R. Leeb, G. Mller-Putz, A. Schlgl, and G. Pfurtscheller. Bci competition iv dataset 2a. [Online]. Available: http://www.bbci.de/competition/iv/#dataset2a

[10] K. K. Ang, Z. Y. Chin, H. Zhang, and C. Guan, "Filter bank common spatial pattern (fbcsp) in brain-computer interface," in *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*. IEEE, 2008, pp. 2390–2397.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[12] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork rnn," in *International Conference on Machine Learning*, 2014, pp. 1863–1871.

[13] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting." *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[14] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[15] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.